

1 **CLAIMS**

2

3

4 1. A method comprising:

5 loading one or more source processing chains to support execution of a

6 development project; and

7 determining whether each of the one or more processing chains will be

8 subsequently required during execution of this or another development project

9 and, if so, caching those filter chains which will be subsequently required.

10

11 2. A method according to claim 1, wherein loading of the one or more

12 processing chains comprises:

13 identifying which source(s) will be required to support execution of the

14 next M seconds of the development project;

15 searching one or more cache(s) to determine whether the source processing

16 chain(s) associated with the source(s) are available within the one or more

17 cache(s); and

18 retrieving the one or more processing chains from a memory location

19 denoted by an associated one or more pointers in the cache for integration with the

20 development project.

21

22 3. A method according to claim 2, further comprising:

23 determining whether processing chain(s) retrieved from the cache(s) satisfy

24 processing requirements of the development project; and

25

1 modifying one or more objects of one or more of the processing chain(s)
2 retrieved from the cache(s) that do not satisfy the processing requirements of the
3 development project.

4
5 4. A method according to claim 3, wherein modifying one or more
6 objects may comprise one or more of adding processing objects to the processing
7 chain(s), removing one or more processing objects from the processing chain(s),
8 or changing one or more operating attributes of one or more processing objects
9 within the processing chain(s).

10
11 5. A method according to claim 2, wherein M is at least as long as
12 necessary to construct a processing chain.

13
14 6. A method according to claim 1, wherein determining whether a
15 processing chain will subsequently be required comprises:

16 determining whether any future calls to a source coupled to the processing
17 chain exist within this development project; and

18 determining whether any future calls to a source coupled to the processing
19 chain may be received during execution of future development projects.

20
21 7. A method according to claim 6, wherein it is assumed that each
22 processing chain may well be required to support future execution of this or a
23 future development project.

1 **8.** A method according to claim 1, wherein caching the processing chain
2 comprises:

3 assigning the processing chain a unique identifier; and

4 storing the unique identifier along with a pointer to a memory location
5 occupied by the processing chain in a cache.

6
7 **9.** A method according to claim 8, wherein the unique identifier is one
8 or more of a source file handle, a source file name, a random numeric identifier
9 uniquely assigned to the processing chain, a graphical icon, an alphanumeric
10 character, and the like.

11
12 **10.** A storage medium comprising a plurality of executable instructions
13 which, when executed, implement a method according to claim 1.

14
15 **11.** A computing system comprising:

16 a storage medium having stored therein a plurality of executable
17 instructions; and

18 an execution unit, coupled to the storage medium, to execute at least a
19 subset of the plurality of executable instructions to implement a method according
20 to claim 1.

21
22 **12.** A method comprising:

23 generating a source chain for use in a development project; and

24 caching the source chain when it is not currently required in the
25 development project.

1
2 **13.** A method according to claim 12, wherein the processing chain is
3 cached only if it will subsequently be required in the development project, or a
4 future development project.

5
6 **14.** A method according to claim 12, wherein caching the source chain
7 comprises:

8 generating an identifier which is uniquely assigned to the processing chain;
9 and

10 storing the identifier along with a pointer to memory occupied by the
11 processing chain in a cache of processing chain pointers.

12
13 **15.** A method according to claim 14, wherein the identifier is one or
14 more of a source file handle, a file name, an icon, a randomly generated number
15 uniquely assigned to the processing chain, an alphanumeric identifier, and the like.

16
17 **16.** A method according to claim 12, further comprising:
18 identifying a need for a source processing chain; and
19 retrieving a suitable processing chain from a cache of such processing
20 chains.

21
22 **17.** A method according to claim 16, further comprising:
23 integrating the retrieved processing chain into the development project.
24
25

1 **18.** A method according to claim 16, further comprising:
2 modifying one or more attributes of the retrieved processing chain before
3 integration into the development project.
4

5 **19.** A method according to claim 18, wherein modification to the
6 retrieved processing chain may include one or more of adding processing objects
7 to the processing chain, removing processing objects from the processing chain,
8 altering one or more processing characteristics of one or more processing objects
9 of the processing chain, and the like.
10

11 **20.** A storage medium comprising a plurality of executable instructions
12 which, when executed, implement a method according to claim 12.
13

14 **21.** A computing system comprising:
15 a storage medium having stored therein a plurality of executable
16 instructions; and
17 an execution unit, coupled to the storage medium, to execute at least a
18 plurality of the instructions to implement a method according to claim 12.
19

20 **22.** A system comprising:
21 a plurality of sources; and
22 an interface, selectively coupled to the plurality of sources, to generate and
23 implement a development project of processing chains, wherein the interface loads
24 a processing chain for each of the plurality of media sources at a point during the
25

1 execution of the project when the chain is required, and wherein the interface is
2 configured to unload at least a subset of the chains when they are not required.

3
4 **23.** A system according to claim 22, wherein the interface only loads
5 those processing chains required during the next M seconds of project execution,
6 and if a current chain-count does not exceed a threshold, T.

7
8 **24.** A system according to claim 23, wherein M is less than a time
9 required to load a processing chain.

10
11 **25.** A system according to claim 23, wherein if the currently loaded
12 chain-count has reached a threshold, T, the interface identifies one or more
13 currently loaded chains that can be unloaded.

14
15 **26.** A system according to claim 25, wherein the interface identifies one
16 or more currently loaded chains that will not be used during the next N seconds to
17 unload.

18
19 **27.** A system according to claim 25, wherein the interface determines
20 whether the identified one or more chains will be required during subsequent
21 execution of the project, or in a future project and, if so, caches the identified
22 chain(s).

1 **28.** A system according to claim 27, wherein the interface assigns a
2 unique identifier to processing chains to be cached, and stores the unique identifier
3 along with a pointer to memory wherein the processing chain resides in a
4 processing chain cache.

5
6 **29.** A system according to claim 22, wherein the interface removes the
7 identified chains from the active project and caches the removed chains.

8
9 **30.** A system according to claim 22, wherein the interface loads
10 processing chains by first searching a cache of processing chains for a suitable
11 match.

12
13 **31.** A system according to claim 30, wherein if the interface identifies a
14 suitable match, the processing chain is retrieved from memory for integration
15 within the processing project.

16
17 **32.** A system according to claim 31, wherein the interface modifies one
18 or more attributes of the retrieved processing chain before integration within the
19 processing project.

1 **33.** A system according to claim 32, wherein modifying the processing
2 chain, the interface performs one or more of adding one or more processing
3 objects to the processing chain, removing one or more processing objects from the
4 processing chain, modifying one or more processing characteristics of one or more
5 processing objects within the processing chain.